**Introduction:**

The security of public enclaves, isolated and secure parts of memory intended to thwart a variety of attacks, in particular memory corruption, is investigated in this paper. Specifically, our focus is on the host-to-enclave boundary that is crucial in guaranteeing the integrity of enclave code. With this boundary in mind, the paper describes how APIs are implemented at this boundary, provides common vulnerabilities at this boundary, and suggest means for mitigation. The background of this work is the use of trusted execution environments, such as Intel SGX, to provide robust security guarantees for sensitive data and computation.

**Research Question:**

The primary research question addressed in this paper is: What are the common vulnerabilities at the host to enclave boundary, how secured are public enclave against memory corruption attack.

**Research Methodology:**

The research methodology consists of several key components:

Analysis of Public Enclaves:

To carry out this objective, the paper first provides a detailed evaluation of public enclaves, especially those based on Intel SGX, in terms of design and security. This is because it concerns issues such as the evaluation of the encryption and integrity protection offered by these enclaves.

Host-to-Enclave Boundary Examination:

A detailed analysis is carried out on the host-enclave boundary to identify these interface points and the implementation of the APIs. The study aims to discuss frequent mistakes as well as mistakes that would lead to memory corruption.

Threat Model Assessment:

This paper evaluates the threat model of SGX and undertakes a deeper analysis of how it works in the case of using the Azure SGX-based enclaves with Microsoft. This means analyzing the possibilities of bypassing methods applied by SGX for memory protection against different types of attacks.

Pointer Handling and Data Management:

The impacts of these primitives are discussed in the context of how enclaves deal with pointers and data: enclaves need to copy input data outside enclaves. This copying mechanism is checked as to its susceptibility to being exploited by attackers.

Attack Analysis:

Control flow manipulation, data corruptions or manipulation, and return-oriented programming attacks are then discussed. The paper makes a vignette of present physical defense mechanisms and their deficiency vis-a-vis the advanced ROP attacks such as the Dark-ROP.

**Results:**

The study’s findings highlight several critical points:

Vulnerabilities at the Host-to-Enclave Boundary:

The major source of memory corruption vulnerabilities at the host-to-enclave boundary are identified as erroneous API implementations. All of these errors result in the inflation of counterfeit objects within the enclave’s secure memory, compromising the way the enclave is designed to protect itself.

Effectiveness of SGX Protections:

SGX offers high quality encryption and integrity checks, but none of these are foolproof. The paper argues that current randomization schemes on SGX do not fully shield against all forms of ROP attacks and that publishers' enclaves are therefore susceptible to sophisticated exploits.

Challenges with Pointer Handling:

Because enclaves must copy input data from outside the enclave, they can be subject to manipulation by attackers. Even after SGX SDK updates to support memory operations, these are not sufficient for processing of complex data types, which makes them vulerable.

Persistent Threat of Non-Control Data Attacks:

The paper also points out however that non control data attacks still pose a very serious threat. Despite data execution prevention, attackers can manipulate data to affect program execution without having to inject malicious code.'

**Analysis of Results:**

The analysis of the results reveals several important insights:

API Implementation Pitfalls:

Important lessons for careful API implementation at the host to enclave boundary are underscored by the findings. This area is rife with errors, which is a primary source of memory corruption vulnerabilities; we must validate and test more rigorously.

Limitations of Current Defense Mechanisms:

While SGX and other TEE technologies have made impressive progress, existing mechanisms are insufficient to withstand all types of memory corruption attack. In particular, advanced ROP attacks have traditionally proved to be a persistent challenge for which necessitates the further development of a more sophisticated protection schemes.

Necessity for Comprehensive Protection:

This indicates that a holistic approach to enclave security is needed to deal with the persistent threat of non control data attacks. Nothing can be taken for granted, and all potential vulnerabilities need to be considered critical and defense mechanisms need to be designed to provide a broad attack vector.

**Conclusion:**

This paper is concerned with a detailed analysis of public enclave memory corruption vulnerabilities at the host to enclave boundary. Significant security concerns, and an urgent need for improved API implementation and better overall protection mechanisms are provided in the findings. Finally, the paper addresses these vulnerabilities to strengthen TEE security and provide guidance on future research and development in this important area.